IN THE CLAIMS:

Claims 1-14 are attached "as-is".

Please amend claim 15, as indicated.

1. (Original). In a computer system executing a Java application and a hierarchical database, a method for providing data access from said Java application to said hierarchical database, said method comprising the steps of:

(a) determining a JDBC type for said Java application to represent a column in a result set on said hierarchical database;

(b) determining a Java class name for said Java application to represent said column;

(c) creating a Java ResultSet object to represent the data in said column and determining a metadata value for said Java application result set to represent said column;

(d) closing embedded result set objects for said result set.

2. (Original). The method as in Claim 1 wherein said step of determining said JDBC type to represent said column further includes the steps of:

(a) determining if said column contains hierarchical data, and if so;

(b) setting said JDBC type to "Java.sql.Types.OTHER";

(c) determining, in step (a) above, that said column does not contain hierarchical data;

(d) handling said column in a normal fashion for non-hierarchical data.

3. (Original). The method as in Claim 1 wherein said step of determining said JDBC type to represent said column further includes the steps of:

(a) determining if said column contains hierarchical data, and if so;

(b) setting said Java class name type to "Java.sql.ResultSet";

(c) determining in step (a) above that said column does not contain said hierarchical data; and,

(d) handling said column in a normal fashion for non-hierarchical data.

4. (Original). The method as in Claim 1 wherein said step (c) of determining said metadata value to represent said column further includes the steps of:

(a) determining if said column contains hierarchical data, and if so;

(b) creating a result set using said hierarchical data as contents;

(c) determining that said column of step (a) above does not contain hierarchical data and if so;

(d) handling said column in a normal fashion for non-hierarchical data.

5. (Original). The method as in Claim 4, which further includes the steps of:

(a) determining if a result set metadata exists for said column, and if so;

(b) creating and populating a result set metadata object for said column;

(c) adding said result set metadata object to a metadata collection;

(d) determining in step (a) above that said result set metadata does not exist for said column;

(e) making said result set to reference said existing result set metadata.

6. (Original). The method as in Claim 1 wherein said step of closing said embedded result set objects for said result set further includes the steps of:

(a) determining if said result set contains embedded result set objects, and if so;

(b) closing each of said results set objects;

(c) determining in step (a) above that said result set does not contain said embedded result set objects, and if so,

(d) closing said result set in a normal fashion.

7. (Original). A storage medium encoded with machine-readable computer program code for providing data access from a Java application to a hierarchical database, wherein, when the computer program code is executed by a computer, the computer performs the steps of:

(a) determining a JDBC type for said Java application to represent a column in a result set on said hierarchical database;

(b) determining a Java class name for said Java application to represent said column;

(c) constructing a result set object to contain the hierarchical data and determining a metadata value for said Java application to represent said column;

(d) closing embedded result set objects for said result set.

8. (Original). The method as in Claim 7 wherein said step of determining said JDBC type to represent said column further includes the steps of:

(a) determining if said column contains hierarchical data, and if so;

(b) setting said JDBC type to "Java.sql.Types.OTHER";

(c) handling said column in normal fashion, when it is determined in step (a) above that said column does not contain said hierarchical data.

9. (Original). The method as in Claim 7 wherein said step of determining said JDBC type to represent said column in a result set further includes the steps of:

(a) determining if said column contains hierarchical data, and if so;

(b) setting said Java class name type to "Java.sql.ResultSet";

(c) handling said column in normal fashion when it is determined in step (a) above that said column does not contain said hierarchical data.

10. (Original). The method as in Claim 7 wherein said step (c) of determining said metadata value to represent said column in a result set further includes the steps of:

(a) determining if said column contains hierarchical data, and if so;

(b) creating a result set using said hierarchical data as contents;

(c) handling said column in normal fashion, when it is determined in step (a) above that said column does not contain said hierarchical data.

11. (Original). The method as in Claim 10, further including the steps of:

(a) determining if result set metadata exists for said column, and if so;

(b) creating and populating a result set metadata object for said column;

(c) adding said result set metadata object to a metadata collection;

(d) making said result set reference said existing result set metadata, when it is determined in step (a) above that said result set metadata does not exist for said column.

12. (Original). The method as in Claim 7 wherein said step of closing said embedded result set objects for said column further includes the steps of:

(a) determining if said result set contains embedded result set objects, and if so;

(b) closing each of said embedded result set objects;

(c) handling said result set in normal fashion, when it is determined in step (a) above that said result set does not contain said embedded result set objects.

13. (Original). A System for enabling a Java application to utilize Java Data Base Connectivity (JDBC) to access data in a hierarchical database comprising:

(a) CPU means for utilizing a Java application and JDBC driver to access a server means;

(b) said server means for executing JDBC software and Object Linking and Embedding Database (OLE DB) data provider means;

(c) hierarchical data storage means which is enabled for data access via said server means, said hierarchical data storage means holding data in result sets of rows where each row includes a column of data.

14. (Original). The system of claim 13 wherein said CPU means includes:

(a1) means to determine the JDBC type for said Java application which represents a column in a result set on said hierarchical database;

(a2) means to determine that Java class name in said Java application which represents said column;

(a3) means for establishing a metadata value for said Java application which represents said column.

15. (Currently Amended). A system for enabling a Java application for utilizing of JDBC to access data in a hierarchical database comprising:

(a) first CPU means (14) utilizing an Internal Browser (10) for accessing an application server (17);

(b) said application server (17) communicating with a first JDBC driver (18);

(c) second CPU means (15) utilizing a Java application means for communicating with said JDBC driver (18);

(d) third CPU means utilizing a Java application (12) and a second JDBC driver (13) for communicating with said first driver (18);

(e) OLE DB data provider [[mans]] means (19) for connecting said first JDBC driver (18) to an NT Server means (20);

(f) storage means for holding data in a hierarchical database.